Data Model Management for Space Information Systems

J. Steven Hughes[*] and Daniel J. Crichton.[†]
Jet Propulsion Laboratory, Pasadena, CA, 91109

Paul Ramirez[‡]
Jet Propulsion Laboratory, Pasadena, CA, 91109

and

Chris Mattmann[§]
Jet Propulsion Laboratory, Pasadena, CA, 91109

The Reference Architecture for Space Information Management (RASIM) suggests the separation of the data model from software components to promote the development of flexible information management systems. RASIM allows the data model to evolve independently from the software components and results in a robust implementation that remains viable as the domain changes. However, the development and management of data models within RASIM are difficult and time consuming tasks involving the choice of a notation, the capture of the model, its validation for consistency, and the export of the model for implementation. Current limitations to this approach include the lack of ability to capture comprehensive domain knowledge, the loss of significant modeling information during implementation, the lack of model visualization and documentation capabilities, and exports being limited to one or two schema types. The advent of the Semantic Web and its demand for sophisticated data models has addressed this situation by providing a new level of data model management in the form of ontology tools. In this paper we describe the use of a representative ontology tool to capture and manage a data model for a space information system. The resulting ontology is implementation independent. Novel on-line visualization and documentation capabilities are available automatically, and the ability to export to various schemas can be added through tool plug-ins. In addition, the ingestion of data instances into the ontology allows validation of the ontology and results in a domain knowledge base. Semantic browsers are easily configured for the knowledge base. For example the export of the knowledge base to RDF/XML and RDFS/XML and the use of open source metadata browsers provide ready-made user interfaces that support both text- and facet-based search. This paper will present the Planetary Data System (PDS) data model as a use case and describe the import of the data model into an ontology tool. We will also describe the current effort to provide interoperability with the European Space Agency (ESA)/Planetary Science Archive (PSA) which is critically dependent on a common data model.

I. Introduction

The Reference Architecture for Space Information Management[3] suggests the separation of the data model from software components to promote the development of flexible information management systems. The proposed architecture allows the data model to evolve independently from the software components and results in a robust implementation that remains viable as the application domain and technology base evolve over time. The author's experience with the *Planetary Data System (PDS) Catalog* has proven that maintaining a domain data model

[*] Principal Computer Scientist, Section 387, 4800 Oak Grove Dr., 169-315, Grade for first author.
[†] Manager, Department Name, Address/Mail Stop, and AIAA Member Grade for second author.
[‡] Staff Computer Scientist, Department Name, Address/Mail Stop, and AIAA Member Grade for third author.
[§] Staff Computer Scientist, Department Name, Address/Mail Stop, and AIAA Member Grade frth author (etc).

independent from the implemented technology-base sustains a viable operational system while simultaneously addressing both incremental domain changes and significant, often abrupt, technology changes. For example, the PDS Catalog was released as a centralized database application in March, 1990 and was soon re-implemented using a client/server architecture.With the advent of the web, it was then re-hosted as CGI-script web application. Recently two semantic web applications have been prototyped, one using a text-based search engine to provide a "Google-Like" search capability and another using a Semantic Web knowledge-base and metadata browser supporting facet-based search. During period of time, continuous incremental changes to the data model were managed with minimal impact on the operational system.

The development and management of a data model is a difficult, continuous, and time consuming task. For example, the planetary science data model required the significant involvement of several domain and information technology experts and took over three years to develop. Change requests were almost immediately submitted after the model's initial development and continue to the present.

Typically, a data model is captured and documented in a tool using a selected notation and formally validated for consistency. It will then often have to be transformed to a different model for implementation. Popular notations for capturing the data model are Entity-Relationship (ER) and the Unified Modeling Language (UML). Common implementations include the relational model and recently XML schemas.

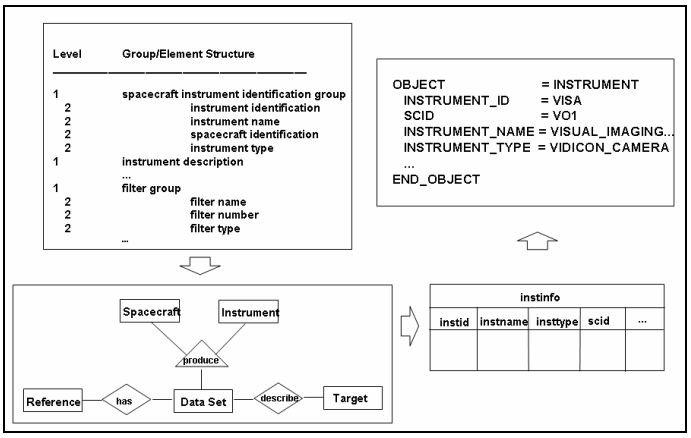


Figure 2. Data Model Transformation Sequence. *Orginally documented using hierarchical structure diagrams, the data model was converted to Entity-Relationship diagrams, a relational schema, and then expressed in the archive in ODL.*

This traditional approach to data model development can have many limitations however, including the inability to capture comprehensive domain knowledge, the loss of significant modeling information during implementation, the lack of model visualization and documentation, and the ability export to only one or two schema types. Modern tools address these limitations but not always completely or consistently.

Semantic Web technology starts to address this situation by providing a new level of data model management in the form of ontology tools. Primarily a result of artificial intelligence research, ontology tools are providing powerful, implementation independent means to capture and manage data models.

In the following sections we will provide a overview of the PDS data model. This will be followed with a description of the data model's import into an ontology tool and the resulting benefits. The deployment of a knowledge-base and semantic browser will be described and finally future work related to international inter-operability will be mentioned.

II. The Planetary Data System Data Model

The Planetary Data System (PDS) data model was developed in the late 1980's to model the various entities and relationships of interest within the Planetary Science Community. It was developed to both prescribe the metadata to be collected for the planetary science data archive and to design the data set catalog, a high level inventory of the data holdings in the archive. The PDS data model is abstractly illustrated in Fig 1.

The data model was originally captured using structure diagrams to identify the key entities, their attributes, and their relationships in the planetary science domain. The essential modeling information was then captured in Entity-Relationship (E-R) diagrams and the attributes were defined as data elements in the Planetary Science data dictionary. The initial model contained over 70 entities and 1200 data elements and focused on the "high-level" entities of the domain such as data sets, instruments, and targets.[15] For the implementation of the high-level data set catalog, the E-R diagrams were transformed into a relational schema. The resulting database supported over 80 sophisticated constraint-based searches for data sets and the other entities.

An object-based language, the Object Description Language (ODL), was created to capture the metadata for loading into the catalog. Simultaneously, the PDS data model was extended to include data products and ODL was used to create product labels for inclusion in the archive. This sequence of model transformations, from structure

diagrams to its representation in ODL is illustrated in Fig 2. Currently all metadata in the PDS archive is captured in ODL and stored in the PDS archive with the science data.

Changes to the data model have been requested continuously since the initial implementation. Most have been small incremental changes that have caused minimal impact on the operational system. However, one major "streamlining" of the model simplified some aspects of the model by move information from data elements to text. It was felt that the work required for the capture, model fitting, and management of individual instrument attributes was not worth the benefits attained for catalog search.

The PDS model continues to evolve today especially at the detailed-level. As new instruments are designed and flown, new data products and additional data elements for describing the products are requested by data providers. In addition, new entities are periodically added to the model to enhance the system, such as a general "resource" entity for describing web resources. The model currently contains over 70 entities and over 1500 data elements.[16] The maintenance of the data model is estimated to require the equivalent of at least one full time staff member. Staff members maintaining the model typically require at least two years prior experience working with the data model.

III. The Planetary Data System Ontology

The PDS Ontology was developed to capture in a single knowledge-base, all known information about the PDS Data Model. This information was extracted from the operational PDS Catalog, Planetary Science Data Dictionary, the PDS Standards Reference, as well as original design documents, development staff notes, and staff memories. This information covered a time span of over 20 years.

A. Background

The PDS data model guides the capture of the metadata necessary to describe the data in the archive and ensure that the data are scientifically useful for current and future scientists. The metadata and science data are collected and validated against the model and after peer review, place in the archive and distributed to the planetary science community.

The PDS data model has two major divisions: (1) the *high-level* division used to implement the PDS Catalog, primarily an inventory of all data holdings at the data set level, and (2) the *detailed or product level* division that describes individual data product types in the archive. The high-level division of the model is the more static part of the model and relatively small containing about 1200 Data Set and tens of thousands of Target instances. The detail-level division of the model is much more dynamic with new product types being added continuously. For example the Cassini project will add at least 121 new product types to the archive.

When the PDS Catalog was implemented the Planetary Science Data Dictionary (PSDD) was also implemented in the catalog database to capture the data model. The PSDD consists of about 25 tables that capture both object and data element definitions and their relationships. User access to the PSDD is through periodically generated hardcopy documents and a web interface.

Much of the original modeling information, such as relationship cardinalities and general descriptive information was never captured in the PSDD and the original design documents remained the only source for this information. In fact the entity relationships are currently captured only in database queries and referential integrity checks. Because of this situation, modifications to the model can only be accomplished by staff members with knowledge of relational databases combined with extensive historical knowledge of the model and its original intent. Individuals with relational database experience are relatively easy to come by however individuals with knowledge of the original design and intent of the model are getting fewer. So it is obvious that the PDS data model be comprehensively documented as soon as possible.

This situation is compounded exponentially when the detailed-level of the model is considered. Typically a product type is designed in the early stages of a mission. However the product design documents typically remain with the mission project and in fact seldom represent the final design used to create the data product instances. Only the actual product instances are placed in the archive.

B. Ontology Development

"An ontology is the product of an attempt to formulate an exhaustive and rigorous conceptual schema about a domain. It is typically a hierarchical data structure containing all the relevant entities and their relationships and rules within that domain (e.g., a domain ontology)"[4]. Since the PDS data model describes all the relevant entities and their relationships and rules within the planetary science archive domain, it is a domain ontology. What remains is the compilation of the information into a tool that provides the standard features and management capabilities that are typically associated with an ontology. The ontology tool, Protégé[11], developed by Stanford Medical Informatics, was chosen to compile the modeling information into a complete ontology.

The major elements for creating the PDS ontology existed in the PSDD, resident in the PDS Catalog database. A program was written to extract the data from the relational tables, formulate object and attribute definitions, and write the definitions to files that can be imported into Protégé. Since Protégé supports a number of database formats, the two most common formats, the native "frames" and RDF formats were chosen.

The PDS objects were modeled in Protégé in two distinct ways. The first is analogous to the model used by the PSDD, namely describing an object in terms of its attributes and sub objects. The PSDD also indicates whether the attributes and sub objects are required or optional. Little additional cardinality information is provided. This approach provides current PDS users with the familiar "definitional" model for object definition. It results in one ontology class and one instance per object. The ontology class is shown in Fig 3.

The second model is a traditional class hierarchy. This approach results in an ontology class for each PDS object. The PDS classifies objects as generic or specific, depending on the existence of optional elements in the definition. Objects are also classified as catalog or product type. An example of the Data Set class is shown in Fig 4.

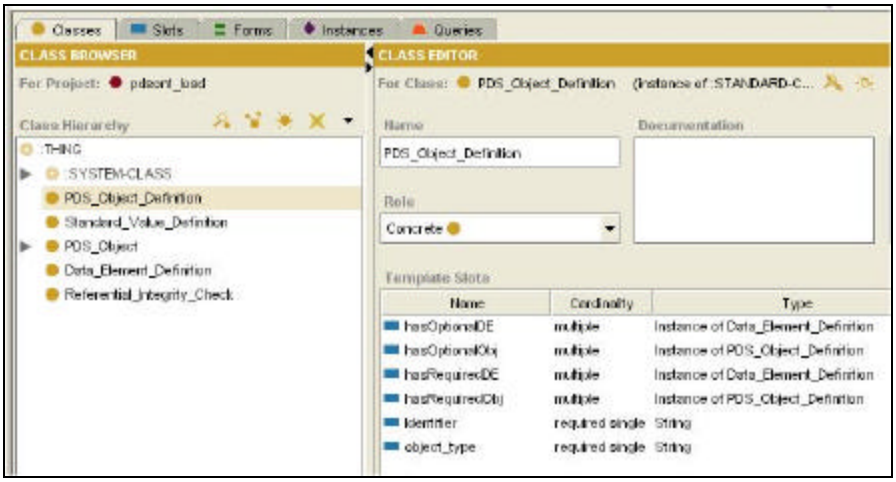


Figure 3. PDS Object Definition Class. *A PDS Object is defined has having data elements and sub objects. Required and optional inclusion are allowed.*

The majority of the information in the PSDD consists of data element definitions. A data element is defined using over 30 attributes. Data elements are modeled in the ontology also using two models. Similar to the object definition class, the single data element definition class seen in Fig 5 is used to define the over 1500 data elements currently in the PSDD. Those data elements used as attributes of object classes in the object hierarchy are each defined using the manner illustrated in Fig 4.

Some entity relationship information was available from the object information extracted to create the object class hierarchy. For example, the PSDD includes an object hierarchy table that can be mined to determine objects and sub objects. However, this does not include domain entity relationships such the targets and missions associated with a data set. Another PSDD table containing checks for referential integrity was used to extract most of these additional relations. However some missing information, such as relationship cardinality, had to be captured from database foreign key constraints, database application SQL joins, external documentation, and staff memory. Figures 3, 4, and 5 also illustrate some of this information including subclass relationships and relationship cardinality.

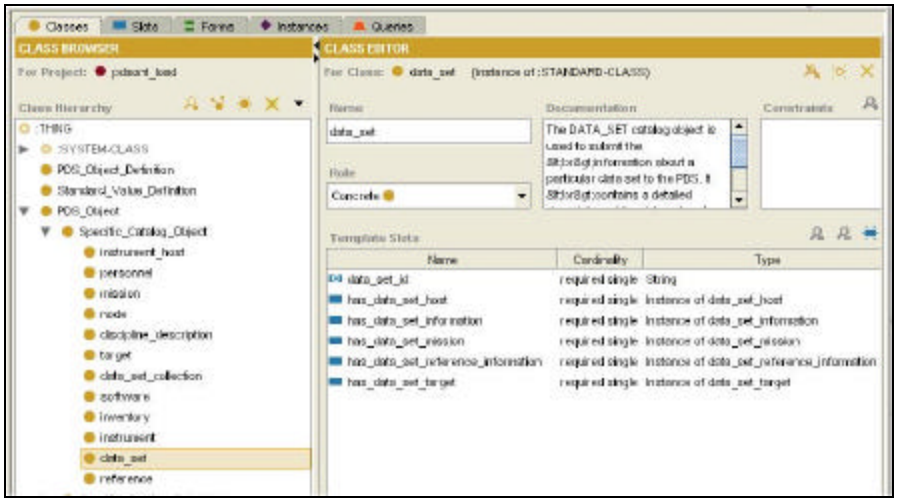


Figure 4. Data Set Class. *The PDS Data Set class is defined as having one required data element and five sub objects.*

Another data model is used by the current PDS distribution subsystem to normalize all PDS object classes in order to support distributed search. This model includes information architecture concepts from the Object-Oriented Data Technology (OODT) project.[1,2] OODT includes broad-scope profile attributes and their relationships that support interoperability across domains[,17]. This model has also been used to load metadata into text- and facet-based search engines to support semantic web applications.[13] And finally a fourth data model has been abstracted out of the implementation model by removing relational artifacts such as tables that implement many-to-many relations. This last model closely reflects the model as captured in the original design. Both of these models have been included in the ontology and provide a seamless integration between the four different but closely related aspects of the PDS data model.

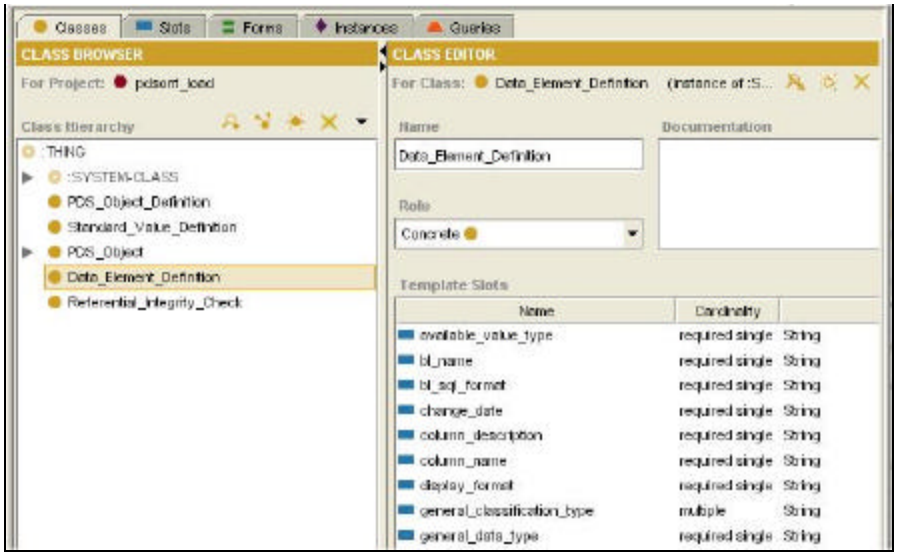
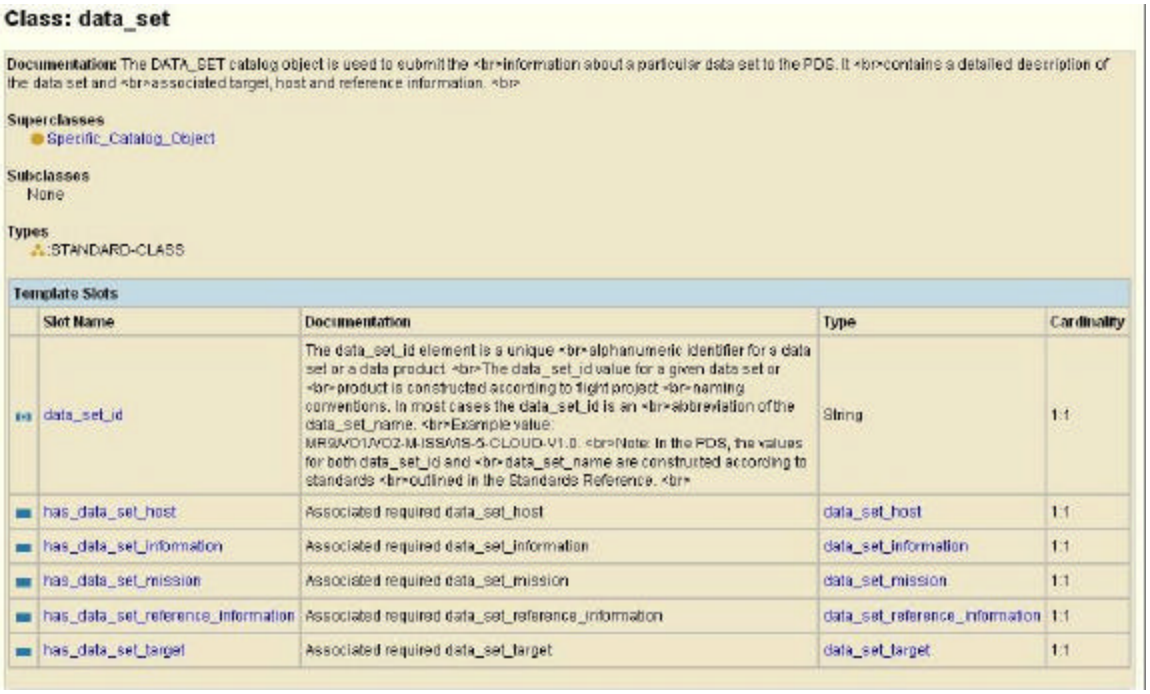


Figure 5. Data Element Definition Class. *The Data Element Definition class defines a data element using several attributes including a long name, short name, data type, and two classification fields .*

C. Documenting and Visualizing the Ontology

The capture of the data model as an ontology has resulted in a more formal and richer specification of the PDS data model. The Protégé ontology tool and various plug-ins provide numerous documentation and visualization tools to view the model. For example, Fig. 6 illustrates the HTML view of the data set class definition. This report format



Class: data_set

Documentation: The DATA_SET catalog object is used to submit the
information about a particular data set to the PDS. It
contains a detailed description of the data set and
associated target, host and reference information.

Superclasses
- Specific_Catalog_Object

Subclasses
None

Types
- :STANDARD-CLASS

Template Slots

	Slot Name	Documentation	Type	Cardinality
	data_set_id	The data_set_id element is a unique alphanumeric identifier for a data set or a data product. The data_set_id value for a given data set or product is constructed according to flight project naming conventions. In most cases the data_set_id is an abbreviation of the data_set_name. Example value: MRSMO1/MO2-M-ISS/MS-5-CLOUD-V1.0. Note: In the PDS, the values for both data_set_id and data_set_name are constructed according to standards outlined in the Standards Reference. 	String	1:1
	has_data_set_host	Associated required data_set_host	data_set_host	1:1
	has_data_set_information	Associated required data_set_information	data_set_information	1:1
	has_data_set_mission	Associated required data_set_mission	data_set_mission	1:1
	has_data_set_reference_information	Associated required data_set_reference_information	data_set_reference_information	1:1
	has_data_set_target	Associated required data_set_target	data_set_target	1:1

Figure 6. Data Set Class. *The Data Set is displayed in the HMTL export format.*

is provided as a default and provides Web browser access to the entire ontology, including the class hierarchy and any class instances that have been ingested. For example the PDS data element definitions are accessible as instances from the Data Element Definition Class.

Many Protégé users develop tool plug-ins. For example, an XML Metadata Interchange (XMI) [10] plug-in creates an XMI file that can be imported into many UML-based design tools. Using visualizer tools allows the creation of UML class diagrams such as that illustrated in Fig 7. The diagram illustrates the more abstract PDS model with relational artifacts removed.

Numerous other plug-ins provide novel graphical interfaces, some that allow user interaction. The screen capture in Fig. 8 illustrates the data set/target relation being high-lighted after rolling the cursor over the line linking the data set and target entities. In the case of complex models, these interfaces allow user control of the through filtering, drill downs, and control of icon color and shape.

Finally, operational schemas can be implemented directly from the ontology, since essentially all aspects of the data model have been captured. For example, the RDF export writes modeling information to a RDFS/XML file. If class instances were ingested into the ontology, these instances are written to a RDF/XML file. Several RDF triple engines now exist that allow query against the resulting RDF knowledge based, after loading. Other plug-ins can and have been written to write additional implementation models such as XML Schema.

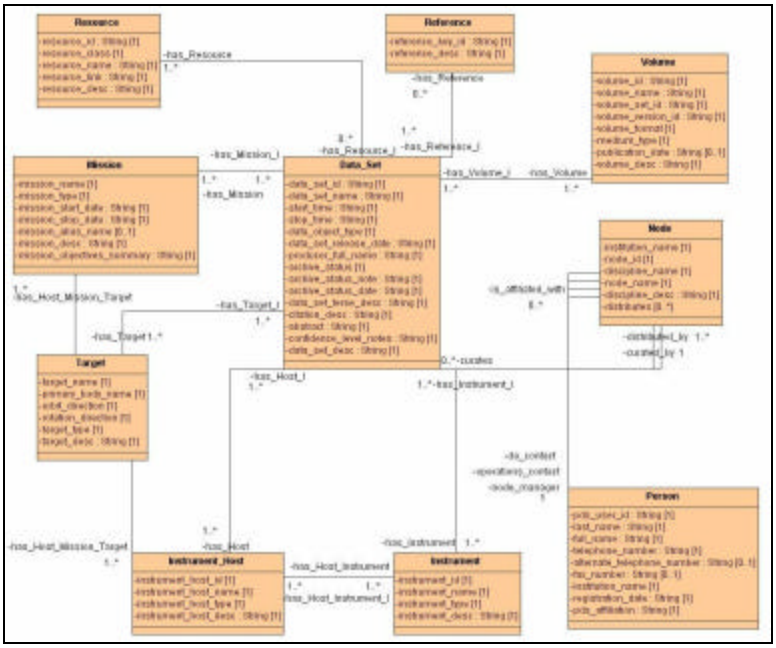


Figure 7. UML Diagram. *A UML class diagram resulting from an XMI export from the ontology.*

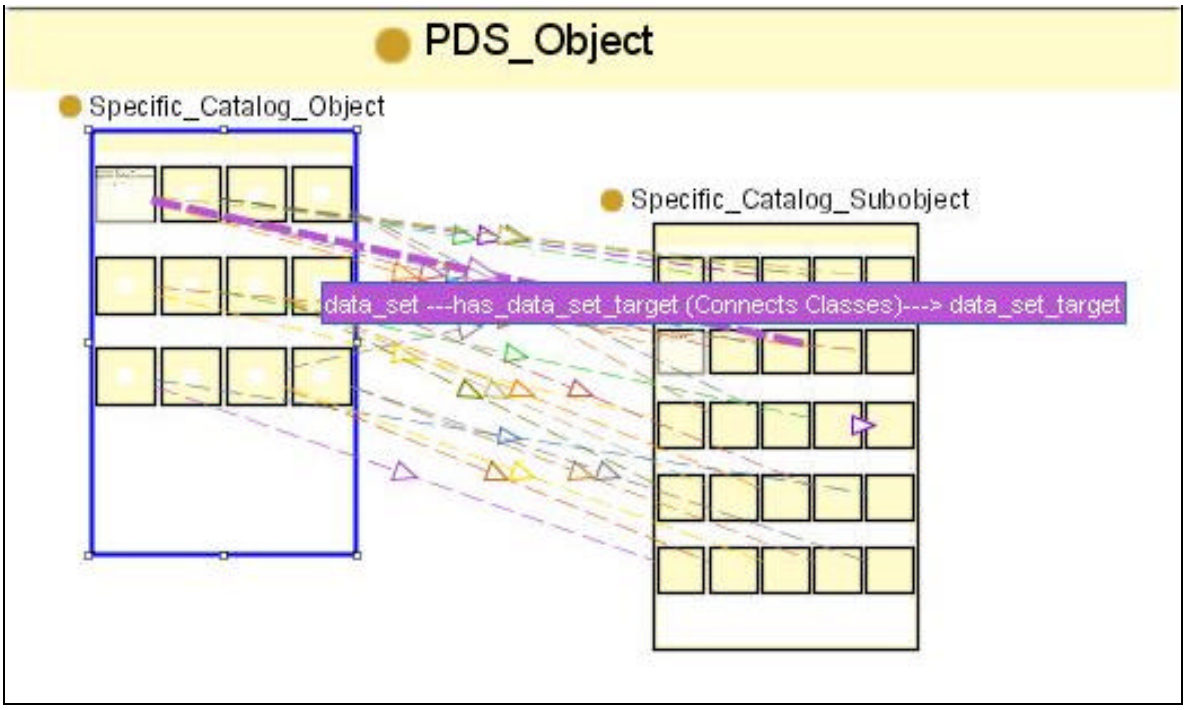


Figure 8. Interactive Graphical Interface. *The data_set_target arc is highlighted showing the relationship between the Data Set object and the Data Set Target subobject.*

IV. Browsing the Ontology

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation[6]". Semantic web technology now available provides powerful tools for capturing and browsing semantically rich knowledge bases. For example the Resource Description Framework (RDF) provides a mechanism for expressing information so that it can be understood by both machines and people. Open source semantic browsers provide powerful text- and facet-based search capabilities for these knowledge bases. In the following we will describe the configuration of a semantic browser for accessing the PDS ontology.

A. Resource Description Framework

The Resource Description Framework (RDF) is a language for representing information about resources in the World Wide Web[5]. RDF is based on the idea of identifying things using Web identifiers (called Uniform Resource Identifiers, or URIs), and describing resources in terms of simple properties and property values. This enables RDF to represent simple statements about resources as a graph of nodes and arcs representing the resources, and their properties and values.

RDF does not provide a mechanism for describing properties or for describing the relationships between these properties and other resources. This is done using the RDF vocabulary description language, RDF Schema[7].

The Protégé tool allows the export of its database content to RDF. In Fig. 9, a snippet from the RDFS/XML file exported from the PDS ontology shows a portion of the definition of the "data element definition" class. The "Class" element defines the Data_Element_Definition class and the next two "Property" elements provide the name and description of the data element.

```
<rdfs:Class rdf:about="&rdf_;Data_Element_Definition"
  rdfs:label="Data_Element_Definition">
  <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdf:Property rdf:about="&rdf_;column_name"
  rdfs:label="column_name">
  <rdfs:domain rdf:resource="&rdf_;Data_Element_Definition"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&rdf_;column_description"
  rdfs:label="column_description">
  <rdfs:domain rdf:resource="&rdf_;Data_Element_Definition"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

Figure 9. Data Element Definition Class - RDFS/XML.

In Fig. 10, a snippet from the RDF/XML file shows a portion of the definition for the data element "data_set_id". This definition is an instance of the Data_Element_Definition class. The second statement identifies the instance as "psdd_de:data_set_id" while the first statement declares it to be of the Data_Element_Definition class. The data type, title, and description follow.

```
<rdfs:Class rdf:about="&rdf_;Data_Element_Definition"
  rdfs:label="Data_Element_Definition">
  <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdf:Property rdf:about="&rdf_;column_name"
  rdfs:label="column_name">
  <rdfs:domain rdf:resource="&rdf_;Data_Element_Definition"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&rdf_;column_description"
  rdfs:label="column_description">
  <rdfs:domain rdf:resource="&rdf_;Data_Element_Definition"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

Figure 10. Data Set Id Definition RDF/XML.

B. Semantic Browsers

Several academic research tasks have developed text- and facet-based search engines for RDF/RDFS knowledge bases. The SIMILE Project deals with applying semantic web technologies to digital libraries and providing the capability to browse and search arbitrary RDF datasets[8]. The Simile/Longwell suite includes web-based RDF browsers that allows the user to browse and search arbitrarily complex RDF datasets using different styles including an end-user friendly view (where all the complexity of RDF is hidden) an RDF-aware view (where all the details are shown). For the PDS ontology browser, the Simile/Longwell suite was chosen to provide facet-based search. Lucene, included in the suite, provides text-based search[9].

The look and feel of the Longwell semantic search application is relatively simple. It is accomplished by creating a set of configuration files that specify an exported RDFS/XML file as the ontology, the RDF/XML file as data, and

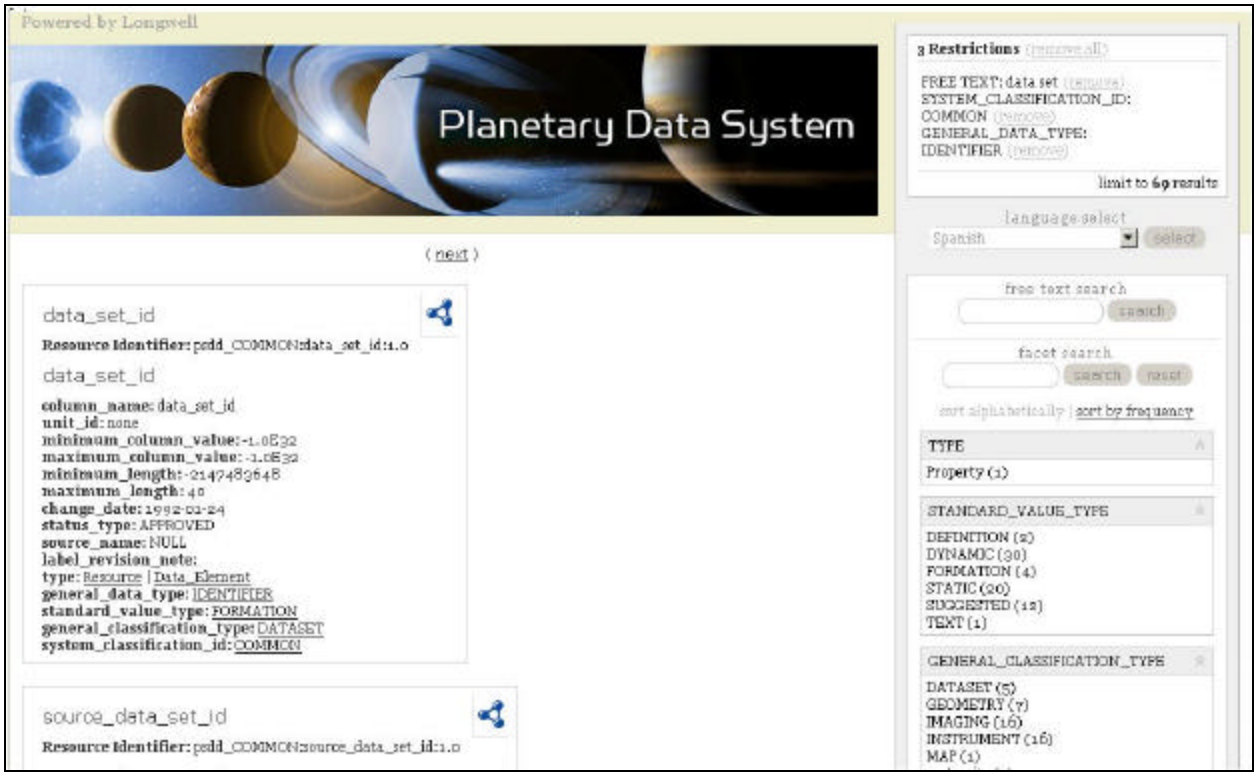


Figure 11. Longwell Metadata Browser. *Results of facet-based restrictions COMMON and IDENTIFIER and text-based restrictions "Data Set".*

the object attributes and values to be used as "facets". The build process produces a Java .war file for deployment as a web application. Fig. 11 illustrates the user interface where users interact by restricting searches using an arbitrary combination of text constraints and facet selection.

The example in Fig. 11 displays the data element definitions resulting after three restrictions. Two restrictions are facet-based restrictions, namely system_classification_id = COMMON and general_data_type = IDENTIFIER. The third is a text-based restriction on the string "data set". Both system_classification_id and general_data_type are PSDD attributes that have been specified as facets for this configuration of the browser. The attributes displayed are controlled by the configuration file. The user can request the display of the complete RDF definition by clicking to the Knowle RDF navigator via the blue triangle. The facet "type", or the ontological class, can also be used as a restriction. The PDS model knowledge base has been configured with all PSDD data element, object, and valid value definitions. Several key attributes, such as data type are specified as facets. This allows very powerful searches to determine, for example, all data elements of type "Character", that are used as properties of the Image object, and that allow the addition of valid values. A search request typically completes in two or three seconds. Typically however large result sets take longer to download.

V. Future Work

The current PDS ontology focused on the high-level entities in the PDS Catalog. In the archive itself, it is estimated that there are over 3000 different product types. To complete the ontology and to support both PDS standards development and ongoing PDS data engineering, a "spider" of the archive to extract each unique product type is being considered. Each product type would be modeled as a unique ontology class. Once accomplished, this would be the first comprehensive documentation of the PDS product types and will be an invaluable tool for the ongoing model development effort. For example, the standards working group will be able to search for the use of data elements in certain contexts to support design decisions for new instrument product types.

VI. Conclusion

The PDS data model, developed over 15 years ago, remains the key component of the PDS archive and its supporting systems. The model has evolved over time making most documentation incomplete or even obsolete. The task to again formally document the model as it currently exists is being accomplished by extracting the modeling information from the operational system, the standards reference document, historical documents, and PDS staff. This information is being organized and modeled formally in an ontology tool. The ontology tool provides the means to generate formal documentation and allow interactive analysis using plug-in tools. The export of the ontology to RDF allows configuration of a knowledge base and access through web-based semantic browsers. Making the PDS data model easily accessible to the wider PDS community will greatly benefit the PDS standards development effort by providing a formal, correct, and current view into the model that resides at the heart of the Planetary Science archive.

Acknowledgments

The authors would like to acknowledge the Jet Propulsion Laboratory Data Systems Standards Program, InterPlanetary Network and Information Systems Directorate, Peter Shames manager, for its financial support.

References

[1] Crichton D, Hughes JS, Hyon J, Kelly S., "Science Search and Retrieval using XML". Proceedings of the 2nd National Conference on Scientific and Technical Data, National Academy of Science, Washington D.C, 2000.

[2] Kelly S, Crichton D, Hughes J.S., "Deploying Object Oriented Data Technology to the Planetary Data System", Proceedings of the 34th Lunar and Planetary Science Conference 1607, 2003.

[3] Consultative Committee on Space Data Systems, "Space Information Architecture", White Paper, Information Architecture Working Group. February 2004, in press.

[4] Wikipedia, The Free Encyclopedia, "Ontology (computer science)", http://en.wikipedia.org/wiki/, August 2005.

[5] RDF Primer, W3C Recommendation, http://www.w3.org/TR/rdf-primer/, 10 February 2004.

[6] The Semantic Web, Scientific American, Tim Berners-Lee, James Hendler, Ora Lassila, May 2001.

[7] RDF Vocabulary Description Language 1.0: RDF Schema W3C Recommendation, 10 February 2004.

[8] The SIMILE Project, http://simile.mit.edu/longwell/index.html.

[9] Apache Lucene, The Apache Software Foundation, http://lucene.apache.org/java/docs/index.html.

[10] XML Metadata Interchange (XMI), Object Management Group, http://www.omg.org/technology/documents/formal/xmi.htm.

[11] Protege, Stanford Medical Informatics, http://protege.stanford.edu/.

[12] Reference Architecture for Space Information Management, Consultative Committee for Space Data Systems, To be Published.

[13] Hughes, J. S., Crichton, D. J., Kelly, S., Mattmann, C., The Semantic Planetary Data System, Ensuring Long-term Preservation and Adding Value to Scientific and Technical data, November 2005.

[15] Catalog Design Document, Planetary Data System, June 1990.

[16] Planetary Science Data Dictionary, Planetary Data System, 2003.

[17] Hughes, J. S., Crichton, D. J., Kelly, S., Mattmann, C., Crichton, J., and Tran, T. Intelligent Resource Discovery using Ontology-based Resource Profiles. *Data Science Journal*, Vol. 4, December 2005.